System Requirements

Specification Index

For

Implementing Django Signal for Logging User Registration

(Topic:- Django Signals)

Version 1.0

Problem Statement Description

Scenario: You have joined a startup company providing an online platform for users to sign up and access various services. To enhance the user registration process and track activity, your task is to create a Django signal that logs information whenever a user successfully registers on the platform. The signal should be triggered automatically after a user is created, logging essential information such as the user's username, email address, and the time the account was created.

This feature will help with auditing and tracking user registrations for further analysis.

Problem Statement

Your Task: Create a Django signal that listens for the post_save event of the User model and logs the user's registration details to a log file. The signal should include:

- **The username.**
- **The email address of the user.**
- **The timestamp when the user registered.**

Instructions:

- **Create a Django signal that listens for the post_save signal of the User model.**
- **The signal should log the user's registration details (username, email, and registration time) to a log file.**
- **Ensure that the signal is only triggered after a new user is successfully created (not updated).**
- **Use Django's logging framework to log the details into a log file. Configure the logger appropriately in the settings.**

Execution Steps to Follow:

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers, need to go to

   Application menu(Three horizontal lines at left top)->Terminal->NewTerminal.

3. The editor Auto Saves the code.
4. If you want to exit (logout) and to continue the coding later anytime(using Save & Exit option on Assessment LandingPage) then you need to use CTRL+Shift+B command compulsorily on code IDE. This will push or save the updated contents in the internal git/repository. Else the code will not be available in the next login.
5. These are time bound assessments the timer would stop if you logout and while

   logging in back using the same credentials the timer would resume from the same time it was stopped from the previous logout.

6. To test any Restful application, the last option on the left panel of IDE, you can find

   ThunderClient, which is the lightweight equivalent of POSTMAN.

7. To test any UI based application the second last option on the left panel of IDE, you can

   find Browser Preview, where you can launch the application.

8. Install ‘djangorestframework’ module before running the code. For this use the following command.

   pip install djangorestframework

9. Use the following command to run the server

   python3 manage.py runserver

10. Mandatory: Before final submission run the following commands to execute testcases

    python3 manage.py test library.test.test_functional

    python3 manage.py test library.test.test_exceptional

    python3 manage.py test library.test.test_boundary

11. To test rest end points

    Click on ‘Thunder Client’ or use Ctrl+Shift+R->Click on ‘New Request’ (at left side of IDE)

12. Once you are done with development and ready with submission, you may navigate to the previous tab and submit the workspace. It is mandatory to click on “Submit Assessment” after you are done with code.

13. You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.